Amendments to the Specification:

[0013] The system allows lossless progressive streaming of 3-D images over the Internet of speed and quality unknown in the prior art. Once a user indicates intention to view an image, a server performs a fast preprocessing step, after which the server quickly responds to requests for viewing specific regions of interest. Each request for an ROI triggers an encoding algorithm, but such that does not involve the full 3-D image. Since the size of the ROI is limited by the size and resolution of the viewing device and by number of frames that can be simultaneously viewed at the client and not by the size of the full 3-D image, only a ~~portion~~ partial progressive coding computation is performed for a local area of the original 3-D data. The same is true for the client computer that performs decoding and rendering only for an ROI and not the ~~fall~~ full 3-D image. The imaging system described herein preferably uses wavelet bases of best performance in streaming ROI data.

[0063] The system workflow is now described with reference to FIG. 1. The user operating a client computer 110 selects, using common browser tools, a 3-D image residing on the image file storage 122. The corresponding URL request is received and processed by the server computer 120. In ~~the~~ case ~~that~~ results of previous computations on the 3-D image are not present in the imaging cache 121, the server computer 120 performs a fast preprocessing algorithm (described herein below as step 2201 with reference to FIG. 22). The result of this computation is inserted into the cache 121. Unlike previous applications or methods, which perform full progressive encoding of the image offline, the goal of the preprocessing step is to allow the server computer 120, after a relatively fast computational step, to serve any ROI specified by the user of the client computer 110. For example, server software processing (step 202) of a sequence of 100 slices 512 by 512 pixels, 16-bit Grayscale each, ~~will~~ typically ~~take~~ takes 1-2 seconds, running on a computer with a Pentium.RTM. processor and the Windows NT.RTM. operating system.

[0064] Once the preprocessing stage is ~~done~~ complete, the server computer 120 notifies the client computer 110 that the image is ready to be served. The server computer 120 also transmits the basic parameters associated with the image such as dimensions, number or resolutions, etc. Upon receiving this notification, the client computer 110 can select any ROI of the 3-D image, as indicated by the user through ~~some~~ a GUI. The ROI is formulated (step 203) into a request list that is sent to the server computer 120. Each such request corresponds to a data block (as described herein below with reference to FIGS. 3 and 8).

[0065] The order of requests in the list corresponds to ~~some~~ a progressive mode selected in the context of the application. For example, the order can be of increasing accuracy. Upon receiving the ROI request list, the server computer 120 processes the requests according to their order. For each ~~such~~ request the server computer 120 checks if the corresponding data block exists in its cache 121. If not, the server computer 120 then computes the data block, stores it in the cache 121 and immediately sends it to the client computer 110. Once a data block that was requested arrives at the client computer 110, it is inserted into its cache 111. At various moments in time during the transfer process, a decision rule invokes a rendering of the ROI. If some of the data blocks required for a high quality rendering of the ROI[[,]] were requested from the server computer 120 but have not arrived yet, the rendering of the ROI will be of lower quality. But, due to the progressive request order, the rendering quality will improve with each received data block as specified. The user can order a change of the ROI, in which case the rendering task at the client computer 110 is canceled. A new request list corresponding to a new ROI is then sent from the client computer 110 to the server computer 120 notifying the server to terminate the previous computation and transmission task and to begin the new one.

[0071] At step 2950, as described below in further detail, 1-D sub-band transform decomposition in [[a]] the z-axis direction is performed upon a portion, such as {HH,HL,LH}, of the two-dimensionally transform-decomposed digital image. The 1-D z-axis decomposition is not performed upon the LL portion in order to preserve its thumbnail qualities.

[0072] At step 2960, the computed transform coefficients are corrected by the performance of integer operations that utilize one or more predetermined correction factors. For reasons described below in further detail, the coefficients are ideally or theoretically to be scaled by a factor or {square root}{square root over (2)} at each resolution level. Nevertheless, the floating-point computations required to scale coefficients by {square root}{square root over (2)} can impact processing speed. Employing integer operations, such as multiplying by integer correction factors, to compensate for the omission of true floating-point operations, maintains processing speed while sacrificing image accuracy at only some ~~of the~~ resolution levels.

[0076] Reference is now made to FIG. 3. The wavelet transform typically creates for each given resolution j eight subbands denoted as lll.sub.j, llh.sub.j, lhl.sub.j, lhh.sub.j, hll.sub.j, hlh.sub.j, hhl.sub.j, hhh.sub.j, while only the subband lll.sub.j is passed to the next resolution level processing. Thus, in the typical case the coefficients of ~~FWT (I)~~ FWT(I) are associated [[to]] with the multiresolution structure shown on FIG. 3. Observe that each coefficient belongs to a ~~certain~~ particular resolution j.

[0078] If a Finite Impulse Response (FIR) subband filter (compactly supported wavelet) is used, each coefficient "influences" only a compactly supported region of the image. High resolution/frequency coefficients are associated with a small region and low resolution/frequency coefficients are associated with a larger region. The coefficients are subdivided into [[the]] tiles as shown in the FIG. 3 for the purpose of efficient rendering. Several subgroups of coefficients are grouped into tiles, associated with the same spatial area, corresponding to the subbands at [[the]] resolution j that are not passed to the next resolution. In some embodiments of the invention, the sub-tiles can have, for example, a length and width of 32 and height of 1, so that the tile contains n x 32^2 x 1 coefficients, where n is the number of sub-bands that are not passed to the next resolution. This relates to the fact that at the time of rendering at the client, the subband tile ~~will provide~~ provides a mean to render a tile of size 64 x 64 2. The parameters tileLength and tileHeight control the size of the subband tiles and can be changed dynamically per 3-D image volume. A 4-D coordinate is associated [[to]] with each such tile, as described by Eq.1.

[0080] Discussion now proceeds to explain how the subband tiles are further decomposed to subband data blocks. The motivation for this finer 5-D decomposition is the following. Each subband tile at the resolution j participates in the rendering of local regions in all the resolutions ≥ j. Assume j corresponds to a very low resolution. ~~It turns out that for~~ For a high quality rendering of a given ROI at the resolution j, only the coefficients with high absolute value are required. Transmitting small coefficients will not have [[any]] a significant visual effect at the time of rendering and thus should be avoided. ~~Also~~ In addition, transmitted coefficients need not be sent at a high precision, since this also will have visual ~~significant~~ significance to the rendering operation. Rather, for each such significant coefficient only a few bits of precision should be transmitted along with the ~~coefficient's~~ sign of the coefficient. Rendering the same ROI at a higher quality or rendering a higher resolution ~~will require~~ requires sending more of the tile's coefficients and also ~~more~~ additional accuracy bits for the coefficients that were already sent. For this reason, each subband tile is further decomposed into "accuracy" data blocks. Each data block is associated with the "bit plane" interval [0,2) of level 0 or the intervals $[2^l, 2^{l+1})$ at level $l \geq 1$, so each data block of the subband tile of Eq.1 will have a 5D coordinate

[0094] An embodiment of the present invention for medical applications minimizes the response time for ROI ~~request~~ requests from the client by evaluating the multi-resolution decomposition in accordance with the following description [[in]] with reference to FIG. 5. The embodiment first employs several steps of HHX-Sub-bands type decomposition (steps 5010, 5020, 5030 and 5040) to obtain thumbnail resolution, which is controlled by the predefined parameter thumbnailResolution. Decomposition is employed on slices Z=Const of the original 3-D image. The embodiment then further employs XXH-Sub-bands type decomposition (steps 5050, 5060 and 5070) to reduce resolution in the Z direction alone. The resulting DC block (step 5070) may be further optionally decomposed by the All-Subbands type decomposition. Decomposition such as depicted in FIG. 5 allows quick calculation of all the thumbnail views of every slice of the study and also swift ("near real time") response to any ROI chosen by the client computer 110.

[0101] The following description is based on the Haar transform, ~~and first~~ initially applied in the X direction. The forward step is the process of coding x(n) to obtain s(n) and d(n). The Haar X direction forward step is given by the following equation.

[0104] Eq 4A shows the following properties of an embodiment of the present invention. First, s(n) comprises scaled LL-subband coefficient. Second, s(n) and $d^{(1)}(n)$ are a de-correlated and reversible couple ([[so]] i.e. they can be transformed back to x(2n) and x(2n+1)), but $d^{(1)}(n)$ is not scaled, so it is assigned half the value ~~use~~ used in similar ~~transformed~~ transforms known in the art. Thus, $d^{(1)}(n)$ is multiplied by 2. Third, ~~and nevertheless,~~ the LSB of the LH-subband coefficient d(n) is known to be 0 and not encoded.

[0107] Eq.4C shows the following properties. First, $d^{(1)}(n)$ and s(n) comprise a de-correlated and reversible couple, but $d^{(1)}(n)$ is not scaled, so it is assigned twice the value ~~use~~ used in similar ~~transformed~~ transforms known in the art. Therefore, $d^{(1)}(n)$ is divided by 2~~. By doing that~~, which causes its least significant bit [[is]] to be lost[[, and]]. This least significant bit cannot be restored, unless this bit is kept as the so called Half-Bit information. This name serves ~~to remind~~

~~us~~ as a reminder that its weight is half that of other coefficients and that it is the least significant (from an approximation point of view).

[0115] 2-D Lossless Transforms have better approximation properties [[to]] of the exact mathematical wavelet transform than previous known separable ~~implementation~~ implementations of the one-dimensional lossless transform.

[0120] Eq.7B scales s(n) and d(n) by a factor ~~off $\sqrt{2}$~~ of $\sqrt{2}$. Because of the fact that all the coefficients resulting from the 2-D Lossless Transforms are scaled properly, all the wavelet coefficients of our entire 3-D transform are scaled in total effect by $\sqrt{2}$ in this case.

[0123] A single one-dimensional lossless wavelet transform ~~doesn't~~ does not preserve the scaling of the resulting coefficients, but the requirement exists, as explained ~~herein above~~ hereinabove in reference to FIG. 5 and 6 for XXH-Subbands type decomposition, to use a one-dimensional transform. The present invention ~~can overcome~~ overcomes this difficulty in the following way: Two different transforms ~~can be~~ are implemented in turn. In one of them, the resulting lowpass coefficients are scaled by a factor of {square root}{square root over (2)}, and in another they are scaled by {square root}{square root over (2)}.sup.-1. This preserves overall scaling of the low pass fraction. In other words, every odd time Eq.7A is used for the forward step and Eq.7C for the inverse step, and every even time the following Eq.7E is used for the forward step, and Eq.7F for the inverse step.

[0124] The resulting scaling factors of wavelet coefficients for all subbands in the proposed multiresolution decomposition are shown on FIG. 7. It can ~~bee~~ be seen that with the present invention all the coefficients are almost equally scaled.

[0125] As for the All-Subbands type decomposition, which is optionally implemented and only for the lowest resolution (or DC) block, the present invention uses the combination of HHX-Subbands and XXH-Subbands types of decompositions.

[0126] Reference is now made to FIG. 8. The figure depicts data blocks as defined in Eq.2. Embodiments of the present invention for lossless transformation [[can]] utilize two additional "half bit planes", one of which is associated with 2-D wavelet transforms on the XY plane as explained in reference to Eq's 3 through 6. The other is associated with a transform in the Z direction as explained in reference to Eq's 7. Each of these two "half bit planes" is implemented as a three-dimensional matrix. For the HHX-Subbands type decomposition this matrix has size (tileLength/2) x (tileLength/2) x tileHeight while for the XXH-Subbands type decomposition this matrix has size tileLength x tileLength x (tileHeight/2).

[0143] 3. ~~Type]~~ Type1: Single coefficient

[0162] As with any progressive subband-coding algorithm, the goal is to efficiently encode the locations of the coefficients that are "exposed" when encoding traverses down to the current bit plane. Naturally, the encoder and decoder have at the time of the scan all the information on what "took place" at the previous higher bit plane scans. As with most methods, the heuristics of the algorithm is that coefficients, which are insignificant at the current bit plane, which are coefficients with absolute value $<\varepsilon 2^b$, are spatially correlated. If it is known that a coefficient is insignificant at the bit plane b then ~~with higher probability so will his neighbors~~ the probability is higher that the neighbors of the coefficient are insignificant as well. This explains why the algorithm groups coefficients into groups of 16 and then 4 and tries to efficiently report (with one symbol) their insignificance. The significance scan uses four binary probability models as listed in the following table.

[0163] The models listed in Table 2 are ~~simply~~ histograms with two levels. They are initialized to equal probabilities at the beginning of each significance scan and are used [[by]] and updated by the arithmetic encoder each time a symbol is encoded. Description now proceeds to details of the significance scan.

[0188] The decoding algorithm is a reversed ~~reflection~~ version of the encoding algorithm performed in the server and described herein above. Decoding is done at the client computer during the progressive rendering operation as described at step 1705 of FIG. 17.

[0195] 5. If the data block (t_x,t_y,t_z,t_resolution,maxBitPlane(t_resolution)) is available at the client, read the value of maxBitPlane (tile), which in an embodiment of the invention is the first byte.

[0196] ~~is available at the client, read the value of maxBitPlane (tile), which in a embodiment of the invention is the first byte.~~

[0197] The outer loop of the decoding algorithm is now described with reference to FIG. 14. The decoding algorithm scans the bit planes from the given b=maxBitPlane(tile) to b=0 (step 1410).

The input to each such bit plane scan is encoded data block (t_x,t_y,t_z, t_resolution ,b) . Since the first stage of the decoding algorithm is arithmetic decoding, at the beginning of each scan, the input module of the arithmetic ~~decoder's input module~~ decoder is redirected (step 1420) to read from the "slot" in the database allocated for the particular data block. There are cases where although a subband tile participates in the rendering of the ROI, some of its data blocks will be missing. A data block could be missing due to the following reasons:

[0208] In such a case, the group ~~surely~~ is certainly significant, it was not encoded and therefore does not need ~~not~~ to be decoded. If the current subgroup remains insignificant, then decoding proceeds to the next subgroup. Else, the decoder removes the label Type4, splits the subgroup to the single coefficients and labels each of them Type1. If the subgroup was not labeled Type4 then it must have been split at a higher bit plane. In both cases the decoder must now continue ~~to handle~~ handling each individual coefficient.

[0216] At the bit plane b the significant coefficient list contains all the coefficients coef(x,y,z) for which b(x,y,z).gtoreq.b at an approximated value. The encoder ~~provided~~ provides this list [[by]] in step 1000 in FIG. 10A, and the decoder handles it in step 2800 in FIG. 28A. When performing this operation, and for each coefficient in the list, the decoder arithmetically decodes the value of test coef(x,y,z)<coef(x,y,z), where coef(x,y,z) is the actual value of the coefficient (not known to the decoder). It then ~~accordingly~~ updates the approximated value accordingly by adding/subtracting from coef(x,y,z) the amount

[0222] Lastly, the parameter progressiveMode determines the order in which data blocks should be transmitted from the server computer 120. The "Progressive By Accuracy" mode is the preferred mode for viewing in low bandwidth environments. "Progressive By Resolution" mode is easier to implement since it does not require the more sophisticated accuracy (bit plane) management and therefore is commonly found in previous solutions. The superiority of the "progressive by accuracy" mode can be proven mathematically ~~proved~~ by showing the superiority of non-linear approximation over linear approximation for the class of real-life images. The "Progressive by Spatial Order" mode is preferred for a cine view of the sequence of 2-D slices one ~~by one~~ after the other. In this mode the image data is ordered and received from the first slice to the last one, such that rendering can be done in parallel [[to]] with the transmission.

[0231] If the RMS increasing factor is greater than 1, it means that the "new RMS" may be greater than a visually negligible error. Thus, the request list should be increased (i.e. more bit-planes should be requested from the server) in order to improve the approximation accuracy. Conversely, if the RMS increasing factor is smaller than 1, the request list can be reduced. The exact specification of this is ~~given~~ provided in the ~~follows~~ following description.

[0232] In step 1702, using the ROI view parameters, the client imaging module calculates a data ~~blocks~~ block request list ordered according to the progressiveMode. Given the parameters worldPolygon, sliceRange and scale it is possible to determine which subband tiles (such as defined in Eq.1) in the frequency domain participate in the reconstruction of the ROI in the image domain. These tiles contain all the coefficients ~~that are~~ required for an Inverse Subband/Wavelet Transform (IWT) step that produces the ROI. The parameter dyadicResolution(ROI), which is the lowest possible dyadic resolution higher than the resolution of the ROI, is first calculated. Any subband tiles of a higher resolution than dyadicResolution(ROI) do not participate in the rendering operation. Their associated data blocks are therefore not requested, since they are visually insignificant for the rendering of the ROI. If $scale \geq 1$ then the highest resolution subband tiles are required. If $scale \leq 2^{1-thumbnailResolution}$ (as described in reference to FIG. 4) then only the lowest resolution tile is required. Mapping is performed for any other value of scale according to Table 5. Table 5 describes the mapping from a given scale to dyadic subband resolution.

[0233] Once it is determined which subband tiles participate in the rendering of the ROI computation proceeds with finding which of their data blocks are visually significant and in what order they should be requested. Using well known rate/distortion rules from the field of image coding, it is possible to determine the preferred order in which the data blocks should be ordered by the client imaging module (and thus delivered by the server). An ordering scheme referred to herein as "Progressive By Accuracy" mode is described in FIG. 19. The mathematical principals of non-linear approximation underlie this approach. ~~First, the~~ The subband coefficients with largest absolute values are requested first since they represent the most visually significant data such as strong edges in the image. Notice that high resolution coefficients with large absolute value are requested before low resolution coefficients with smaller absolute value. Within each given layer of precision (bit plane) (step 1910) where the order of the request is ~~according~~ in accordance with the [[to]] resolution (step 1920), whereby low resolution coefficients are requested first and the coefficients of highest-Sub-band-Resolution are requested last.

[0234] A major difficulty in executing this step is now discussed. Assume a subband tile such as described by Eq.1 is required for the rendering of the ROI. This means that t_resolution≤dyadicResolution (ROI) and the tile is required in the IWT procedure that reconstructs the ROI. There is a need to understand which of the data blocks such as described by Eq.2 associated with the subband tile represent visually insignificant data and thus should not be requested. Sending all of the associated data blocks will not affect the quality of the progressive rendering. However, in many cases transmitting the last [[of]] data blocks associated with high precision is unnecessary since the last blocks will be visually insignificant. In such a case, the user will notice that the transmission of the ROI from the server is still in progress, yet the progressive rendering of the ROI seems to no longer change the displayed image.

[0241] Third, update minRequestPlane accordingly. This rule, that the lower the resolution the less precision is required, originates from the way subband coefficients are scaled according to their resolution. Fourth, ~~wenever~~ whenver deviceDepth=8, the output device is of low resolution. In ~~such a~~ this case the client ~~can request~~ requests less precision. Decrement one bit plane and set

[0243] Last, and according to the description ~~herein above in~~ hereinabove with reference to FIG. 18, additionally reduce or add the number

[0246] In the first example there is given an image depth of 12 ~~bit~~ bits, a screen depth of 8 ~~bit~~ bits and a linear luminance mapping, calculate Maximal_derivative

[0254] Recall that the first four of the five coordinates of a data block (t_x,t_y,t_z,t_resolution,t_bitPlane) denote the subband tile, associated with each data block. From the subband ~~tile's~~ tile coordinates ~~we calculate~~ the dimensions of the "area of visual significance[[.]]" is calculated. That is, the portion of the ROI that is affected by the subband tile.

[0266] Beginning with the lowest resolution of 1, the algorithm executes a multi-resolution march from the first to the last tile of the ROI in the Z direction. The pseudo-code for this recursive march is ~~given~~ presented in FIG. 20. First the allocated matrix is filled with sub-tiles of coefficients calculated on the lower resolution, ~~giving~~ yielding low pass subbands of the current resolution. Then the matrix is complemented by the high pass subbands (wavelet) coefficients, decoded from the database or read from the memory cache as described ~~herein below~~ hereinbelow in reference to FIG. 21. An inverse subband transform described ~~herein below~~ hereinbelow in reference to FIG. 3 produces multiresolution pixels. Each time the highest resolution DyadicResolution(ROI) is reconstructed, it is supplied as input to the resizing and luminance mapping step.

[0274] In step 2203, the server reads from cache or encodes data ~~block~~ blocks associated with low-resolution portions of the ROI, using the cached ~~result~~ results of the preprocessing Step 2201.

[0293] are read from the memory frame buffers and handled one at a time. In Step 2302 the tile is transformed into a tile of size [tilelength , tileLength, tileHeight] containing two types of coefficient data: subband coefficients and "pixels" at a lower resolution. The subband coefficients are processed in step 2303 and step 2304 and are stored in the cache. The lower resolution pixels are inserted in step 2305 [[to]] into a lower resolution memory frame buffer. Whenever ~~such~~ a new sub-tile of lower resolution "pixels" (step 2305) is inserted into a frame, a memory management module performs the following check. If [[the]] a part of the sub-tile exceeds the current virtual boundaries of the memory frame buffer, the corresponding first frames of the memory frame buffer are then considered unnecessary ~~anymore~~. Their memory is (virtually) re-allocated for the purpose of storing ~~the sub-tile's~~ new sub-tile data. The pseudo-code of the memory constraint scan algorithm is detailed in FIG. 24.

[0295] In [[a]] the lossy mode of operation, the low pass filtering can be implemented efficiently in integer numbers without ~~paying~~ much ~~attention to~~ consideration of round-off errors. In lossless mode (when losslessMode=true), care must be taken to ensure that the ~~result~~ results of the low pass step, which are low resolution pixels, can be used in a lossless framework. Specifically, it must be ensured that in a scenario where a user views a low resolution ROI and then zooms into a high resolution ROI, the low resolution data already present at the client side can be used for the rendering such that only the difference between the low resolution and high resolution ~~need~~ needs to be progressively transmitted until lossless quality is reached. Thus, the low pass filter is implemented using a special lossless "integer to integer" transform technique. Unfortunately, this means that this step is slower in lossless mode, due to this special implementation. Nevertheless, it is the embodiment as it provides fast response to the user's first request to view the image in a true lossless efficient imaging system.

[0296] In lossless mode the jumpSize parameter defines the number of lossless wavelet low pass steps that should be ~~done~~ performed. A single low pass step is the same for Haar and CDF (1,3) and for the HHX-Subbands type of decomposition that is the only one used in the "jump". This step is defined by the following Eq.12.